Amendments to the Specification:

Please replace the paragraph beginning on page 1, line 9 with the following:

1. ~~Filed~~ Field of the Invention

The present invention relates generally to the field of message authentication, and more specifically to an authentication implementation which may be applied for cryptography acceleration. In particular, the invention is directed to a hardware implementation to increase the speed at which SHA1 authentication procedures may be performed on data packets transmitted over a computer network.

Please replace the paragraph beginning on page 3, line 6 with the following:

Both MD5 and SHA1 authentication algorithms specify that data is to be processed in 512-bit blocks. If the data in a packet to be processed is not of a multiple of 512 bits, padding is applied to round up the data length to a multiple of 512 bits. Thus, if a data packet that is received by a chip for an authentication is larger then 512 bits, the packet is broken into 512-bits data blocks for authentication processing. If the packet is not a multiple of 512 bits, the data left over following splitting of the packet into complete 512-bit blocks must be padded in order to reach the 512-bit block processing size. The same is true if a packet contains fewer [[then]] than 512 bits of data. For reference, a typical Ethernet packet is up to 1,500 bytes. When such a packet gets split into 512-bit blocks, only the last block gets padded and so that overall a relatively small percentage of padding overhead is required. However for shorter packets, the padding overhead can be much higher. For example, if a packet has just over 512 bits it will need to be divided into two 512-bit blocks, the second of which is mostly padding so that padding overhead approaches 50% of the process data. The authentication of such short data packets is particularly burdensome and time consuming using the conventionally implemented MD5 and SHA1 authentication algorithms.

Please replace the paragraph beginning on page 4, line 16 with the following:

Moreover, the HMAC-MD5-96 and HMAC-SHA1-96 algorithms used in IPSec expand MD5 and SHA1, respectively, by performing two loops of operations. The HMAC algorithm for either MD5 or SHA1 (HMAC-x algorithm) is depicted in FIG. 1. The inner hash (inner loop) and the outer hash (outer loop) use different initial hash states. The outer hash is used to compute a digest based on the result of the inner hash. Since the result of the inner hash is 128 bits long for MD5 and 160 bits long for SHA1, the result must always be padded up to 512 bits and the outer hash only processes the one 512-bit block of data. HMAC-MD5-96 and HMAC-SHA1-96 provide a higher level of security, however additional time is needed to perform the outer hash operation. This additional time becomes significant when the length of the data to be processed is short, in which case, the time required to perform the outer hash operation is comparable to the time required to perform the inner hash operation.

Please replace the paragraph beginning on page 5, line 5 with the following:

Authentication represents a significant proportion of the time required to complete cryptography operations in the application of cryptography protocols incorporating both encryption/decryption and MD5 and/or SHA1 authentication functionalities. In the case of IPSec, authentication is often the time limiting step, particularly for the processing [[or]] of short packets, and thus creates a data processing bottleneck. In particular, of the two algorithms supported by the IPSec protocol, HMAC-SHA1-96 is about twenty-five percent slower than HMAC-MD5-96 in terms of the total computation rounds. Accordingly, techniques to accelerate authentication and relieve this bottleneck would be desirable. Further, accelerated implementations of SHA-1 would benefit any application of this authentication algorithm.

Please replace the paragraph beginning on page 6, line 21 with the following:

In another aspect, the invention pertains to a method of authenticating data transmitted over a computer network. The method involves receiving a data packet stream, splitting the packet data stream into fixed-size data blocks, and processing the fixed-size data blocks using a multi-round authentication engine architecture. The architecture implements hash round logic for a SHA1 multi-round authentication algorithm having a combined adder tree with a timing critical path having a single 32-bit carry look-ahead adder (CLA)<u>.</u> The additions are done in a single clock cycle by rearranging the order of the CLA and circular shift operations.

Please replace the paragraph beginning on page 14, line 7 with the following:

The inner hash is conducted on all 512 bit blocks of a given data packet. The result of <u>the</u> inner hash is 128 bits long for MD5 and 160 bits long for SHA1. The result is padded up to 512 bits and the outer hash processes the one 512-bit block of data to compute a digest based on the result of the inner hash. An output buffer 230 stores the digest and outputs it through a multiplexer 232.

Please replace the paragraph beginning on page 14, line 13 with the following:

As noted above, of the two algorithms supported by the IPSec protocol, HMAC-SHA1-96 is about twenty-five percent slower than HMAC-MD5-96 in terms of the total computation rounds. One way to improve HMAC-SHA1-96 in an IPSec-supporting hardware implementation is to collapse multiple rounds of logic into a single clock cycle thus the total number of clocks required for HMAC-SHA1-96 operation is reduced. The same approach may be applied to any multi-round authentication algorithm. However, simply collapsing the logic for multiple rounds into a single clock cycle can cause the delay to compute the collapsed logic to increase, therefore reducing the maximum clock frequency.

Please replace the paragraph beginning on page 16, line 8 with the following:

As described above, both MD5 and SHA1 algorithms specify that the final hash states of every 512-bit block are to be added together with the initial hash states. The results are then used as the initial states of the next 512-bit block. In MD5, values of four pairs of 32-bit registers need to be added and in SHA1, five pairs. Considering that each 32-bit addition takes one clock cycle, a typical hardware implementation would use four extra cycles in MD5 and five extra cycles in SHA1 to perform these additions if hardware resources are limited.

Please replace the paragraph beginning on page 19, line 10 with the following:

FIG. 8 illustrates a block diagram of a carry look-ahead adder cell. A CLA is designed to reduce the carry propagation delay. It uses specially designed logic to compute carry prior to the summation. ~~[The module~~ The module 'Carry Look-ahead Logic' in FIG. 8 represents the logic that generates C1, C2 . . . based on P and G in the equations that follow. Once the carries are generated, on a per bit basis, there will be the two inputs Ai and Bi as well as the final carry input. The sum (S1, S1) is an XOR result of all three inputs.

Please replace the paragraph beginning on page 22, line 19 with the following:

Logic implementation according to the present invention extends the effectiveness of CSA across both steps. [[Is]] In so doing, the partial results of Step 1 can be saved without applying CLA, removing one CLA from the critical path (CLA is always time-consuming due to the carry propagation). This is achieved by manipulating the position of the circular shift, normally happening at a fixed bit location in SHA1. In accordance with the present invention, the order of the circular shift operation is switched with the last CLA addition (A+B) in Step 1, thereby replacing the CLA operation in Step 1 with more CSAs.